Timed-Release Cryptography

Wenbo Mao
Hewlett-Packard Laboratories
Filton Road, Stoke Gifford
Bristol BS34 8QZ
United Kingdom
wm@hplb.hpl.hp.com

March 7, 2001

Abstract

Let n be a large composite number. Without factoring n, the validation of $a^{2^t} (\bmod n)$ given a, t with $gcd(a, n) = 1$ and $t < n$ can be done in t squarings modulo n. For $t \ll n$ (e.g., $n > 2^{1024}$ and $t < 2^{100}$), no lower complexity than t squarings is known to fulfill this task (even considering massive parallelisation). Rivest et al suggested to use such constructions as good candidates for realising timed-release crypto problems.

We argue the necessity for zero-knowledge proof of the correctness of such constructions and propose the first practically efficient protocol for a realisation. Our protocol proves, in $\log_2 t$ standard crypto operations, the correctness of $(a^e)^{2^t} (\bmod n)$ with respect to a^e where e is an RSA encryption exponent. With such a proof, a *Timed-release RSA Encryption* of a message M can be given as $a^{2^t} M (\bmod n)$ with the assertion that the correct decryption of the RSA ciphertext $M^e (\bmod n)$ can be obtained by performing t squarings modulo n starting from a. *Timed-release RSA signatures* can be constructed analogously.

Keywords Timed-release cryptography, Time-lock puzzles, Timed commitments, Efficient zero-knowledge protocols.

1 Introduction

Let n be a large composite natural number. Given $t < n$ and $gcd(a, n) = 1$, without factoring n, the validation of

$$X \equiv a^{2^t} (\bmod n) \tag{1}$$

can be done in t squarings mod n. However if $\phi(n)$ (Euler's phi function of n) is known, then the job can be completed in $O(\log n)$ multiplications via the following two steps:

$$u \stackrel{def}{=} 2^t (\bmod \phi(n)), \tag{2}$$

$$X \stackrel{def}{=} a^u (\bmod n). \tag{3}$$

For $t \ll n$ (e.g., $n > 2^{1024}$ and $t < 2^{100}$), it can be anticipated that factoring of n (and hence computing $\phi(n)$ for performing the above steps) will be much more difficult than performing t squarings. Under this condition we do not know any other method which, without using the factorisation of n, can compute $a^{2^t} (\bmod n)$ in time less than t squarings. Moreover, because each squaring can only be performed on the result of the previous squaring, it is not known how to speedup the t squarings via parallelisation of multiple processors. Parallelisation of each squaring step cannot achieve a great deal of speedup since a squaring step only needs a trivial computational resource and so any non-trivial scale of parallelisation of a squaring step is likely to be penalised by communication delays among the processors.

These properties suggest that the language

$$L(a, t, n) = \{ (a, t, a^{2^t} \bmod n) \mid t < n,\ gcd(a, n) = 1 \} \tag{4}$$

forms a good candidate for the realisation of timed-release crypto problems. Rivest, Shamir and Wagner pioneered the use of this language in a time-lock puzzle scheme [11]. In their scheme a puzzle is a triple (t, a, n) and the instruction for finding its solution is to perform t squarings mod n starting from a which leads to $a^{2^t} (\bmod n)$. A puzzle maker, with the factorisation knowledge of n, can construct a puzzle efficiently using the steps in (2) and (3) and can fine tune the difficulty for finding the solution by choosing t in a vast range. For instance, the MIT Laboratory for Computer Science (LCS) has implemented the time-lock puzzle of Rivest et al into “The LCS35 Time Capsule Crypto-Puzzle” and started its solving routine on 4th April 1999. It is estimated that the solution to the LCS35 Time Capsule Crypto-Puzzle will be found in 35 years from 1999, or on the 70 years from the inception of the MIT-LCS [10]. (Though we will discuss a problem of this puzzle in §1.2.)

1.1 Applications

Boneh and Naor used a subset of $L(a, t, n)$ (details to be discussed in §1.2) and constructed a timed-release crypto primitive which they called “timed commitments” [3]. Besides several suggested applications they suggested an interesting use of their primitive for solving a long-standing problem in fair contract signing. A previous solution (due to Damgård [6]) for fair contract signing between two remote and mutually distrusted parties is to let them exchange signatures of a contract via gradual release of secrets. A major drawback with that solution is that it only provides a *weak fairness*. Let us describe this weakness by using, for example, a discrete-logarithm based signature scheme. A signature being gradually released relates to a series of discrete logarithm problems with the discrete logarithm values to have gradually decreasing magnitudes. Sooner or later before the two parties completes their exchange, one of them may find himself in a position of extracting a discrete logarithm which is sufficiently small with respect to his computational resource. It is well-known (e.g., the work of Van Oorschot and Wiener on the parallelised rho method [13]) that parallelisation is effective for extracting small discrete logarithms. So the resourceful party (one who is able to afford vast parallelisation) can abort the exchange at that point and wins an advanced position unfairly. Boneh and Naor suggested to seal signatures under exchange using elements in $L(a, t, n)$. Recall the aforementioned non-parallelisable property for re-constructing the elements in $L(a, t, n)$, a roughly equal time can be imposed for both parties to open the sealed signatures regardless of their (maybe vast) difference in computing resources. In this way, they argued that *strong fairness* for contract signing can be achieved.

Rivest et al suggested several other applications of timed-release cryptography [11]:

- A bidder in an auction wants to seal his bid so that it can only be opened after the bidding period is closed.
- A homeowner wants to give his mortgage holder a series of encrypted mortgage payments. These might be encrypted digital cash with different decryption dates, so that one payment becomes decryptable (and thus usable by the bank) at the beginning of each successive month.
- A key-escrow scheme can be based on timed-release crypto, so that the government can get the message keys, but only after a fixed, pre-determined period.
- An individual wants to encrypt his diaries so that they are only decryptable after fifty years (when the individual may have forgotten the decryption key).

1.2 Previous Work and Unsolved Problem

With the nice properties of $L(a, t, n)$ we are only half way to the realisation of timed-release cryptography. In most imaginable applications where timed-release crypto may play a role, it is necessary for a problem constructor to prove (ideally in zero-knowledge) the correct construction of the problem (e.g., without a correctness proof, the strong fairness property of the fair-exchange application is absent).

From the problem's membership in NP we know that there exists a zero-knowledge proof for a membership assertion regarding language $L(a, t, n)$. Such a proof can be constructed via a general method (e.g., the work of Goldrich et al [8]). However, the performance of a zero-knowledge proof in a general construction is not suitable for practical use. By the performance for practical use we mean an efficiency measured by a *small* polynomial in some typical parameters (e.g., the bit length of n). To our knowledge, there exists no practically efficient zero-knowledge protocols for proving the general case of the membership in $L(a, t, n)$. We say so with our awareness of the work of Boneh and Naor of "timed commitments" [3].

Boneh and Naor constructed a practically efficient protocol for proving membership in the subset of $L(a, t, n)$ where $t = 2^k$ with k being any natural number. The time control that the elements in this subset can offer has granularity of powers of 2. This granularity is too coarse. Boneh and Naor envisioned $k \in [30, ..., 50]$ for typical cases in applications. While it is evident that k decreasing from 30 downwards will quickly trivialise a timed-release crypto problem as 2^{30} is already at the level of a small polynomial in the secure bit length of n (usually 2^{10}), a k increasing from 30 upwards will harden the problem in such increasingly giant steps that imaginable services (e.g., the strong fairness for gradual disclosure of secret proposed in [3]) will quickly become unattractive or unusable. Taking the LCS35 Time Capsule for example, suppose that the 35-year-opening-time capsule is in that subset (so the correctness can be efficiently proved with the protocol in [3]), then the only other elements in that subset with opening times close to 35 years will be 17.5 years and 70 years.

The Time-Lock-Puzzle work of Rivest et al [11] did not provide a method for proving the correct construction of a timed-release crypto problem.

1.3 Our Work

We construct the first practically efficient zero-knowledge proof protocol for demonstrating the membership in $L(a,t,n)$ which runs in $\log_2 t$ steps, each an exponentiation modulo n, or $O(\log_2 t(\log_2 n)^3)$ bit operations in total. This efficiency suits practical uses. The membership demonstration can be conducted in terms of $(a^e)^{2^t}(\bmod n) \in L(a^e,t,n)$ on given a and a^e where e is an RSA encryption exponent. Then we are able to provide two timed-release crypto primitives, one for timed release of a message in RSA encryption, and the other for timed release of an RSA signature. In the former, a message M can be sealed in $a^{2^t}M(\bmod n)$, and the established membership asserts that the correct decryption of the RSA ciphertext $M^e(\bmod n)$ can be obtained by performing t squarings modulo n starting from a. The latter primitive can be constructed analogously.

Our schemes provide general methods for the use of timed-release cryptography.

1.4 Organisation

In the next section we agree on notations to be used in the paper. In Section 3 we construct general methods for timed-release cryptography based on proven membership in $L(a,t,n)$. In Section 4 we construct our membership proof protocol working with an RSA modulus of a safe-prime structure. In Section 5 we generalise our result to working with any odd composite modulus which is difficult to factor.

2 Notation

Throughout the paper we use the following notation. Z_n denotes the ring of integers modulo n. Z_n^* denotes the multiplicative group of integers modulo n. $\phi(n)$ denotes Euler's phi function of n, which is the order, i.e., the number of elements, of the group Z_n^*. For an element $a \in Z_n^*$, $Order_n(a)$ denotes the multiplicative order modulo n of a, which is the least index i satisfying $a^i \equiv 1(\bmod n)$; $\langle a \rangle$ denotes the subgroup generated by a; $\left(\frac{x}{n}\right)$ denotes the Jacobi symbol of x mod n. We denote by $J_+(n)$ the subset of Z_n^* containing the elements of the positive Jacobi symbol. For integers a, b, we denote by $gcd(a,b)$ the greatest common divisor of a and b, and by $lcm(a,b)$ the least common multiple of a and b. For a real number r, we denote by $\lfloor r \rfloor$ the floor of r, i.e., r round down to the nearest integer. For an event E, we denote by $Pr[E]$ the probability for E to occur.

3 Timed-Release Crypto with Membership in $L(a, t, n)$

Let Alice be the constructor of a timed-release crypto problem. She begins with constructing a composite natural number $n = pq$ where p and q are two distinct odd prime numbers. Define

$$a(t) \stackrel{def}{=} a^{2^t} (\bmod n), \tag{5}$$

$$a^e(t) \stackrel{def}{=} (a(t))^e (\bmod n), \tag{6}$$

where e is a fixed natural number relatively prime to $\phi(n)$ (in the position of an RSA encryption exponent), and $a \not\equiv \pm 1 \pmod{n}$ is a random element in Z_n^*. Alice can construct $a(t)$ using the steps in (2) and (3).

The following security requirements should be in place: n should be so constructed that $Order_{\phi(n)}(2)$ is sufficiently large, and a should be so chosen that $Order_n(a)$ is sufficiently large.

In the remainder of this section, we assume that Alice has proven to Bob, the verifier, the following membership status (using the protocol in §4):

$$a^e(t) \in L(a^e, t, n). \tag{7}$$

Clearly, this is equivalent to another membership status:

$$a(t) \in L(a, t, n).$$

However in the latter case $a(t)$ is (temporarily) unavailable to Bob due to the difficulty of extracting the e-th root (of $a^e(t)$) in the RSA group.

3.1 Timed-release of an RSA Encryption

For message $M < n$, to make the RSA ciphertext $M^e (\bmod n)$ decryptable in time t, Alice can construct a "timed encryption":

$$TE(M, t) \stackrel{def}{=} a(t)M \pmod{n}. \tag{8}$$

Let Bob be given the tuple $(TE(M, t), a^e(t), e, a, t, n)$ where $a^e(t)$ is constructed in (5) and (6) and has the membership status in (7) proven by Alice. Then from the relation

$$TE(M, t)^e \equiv a^e(t) M^e \pmod{n}, \tag{9}$$

Bob is assured that the plaintext corresponding to the RSA ciphertext $M^e (\bmod n)$ can be obtained from $TE(M, t)$ by performing t squarings modulo n starting from a.

Remark As in the case of a practical public-key encryption scheme, M in (8) should be randomised using a proper plaintext randomisation scheme designed for providing the semantic security (e.g., the OAEP scheme for RSA [1]).

3.2 Timed-release of an RSA Signature

Let e, n be as above and d satisfy $ed \equiv 1 \pmod{\phi(n)}$ (so d is in the position of an RSA signing exponent). For message $M < n$ (see Remark below), to make its RSA signature $M^d (\bmod\, n)$ releasable in time t, Alice can construct a "timed signature":

$$TS(M,t) \stackrel{def}{=} a(t) M^d \,(\bmod\, n). \tag{10}$$

Let Bob be given the tuple $(M, TS(M,t), a^e(t), e, a, t, n)$ where $a^e(t)$ is constructed in (5) and (6) and has the membership status in (7) proven by Alice. Then from the relation

$$TS(M,t)^e \equiv a^e(t) M \,(\bmod\, n), \tag{11}$$

Bob is assured that the RSA signature on M can be obtained from $TS(M,t)$ by performing t squarings modulo n starting from a.

Remark As in the case of a practical digital signature scheme, M in (10) should denote an output from a secure one-way hash function. We further require that the output is in $J_+(n)$. A random padding scheme should make this happen with probability 0.5.

3.3 Security Analysis

3.3.1 Confidentiality of M in $TE(M,t)$

We assume that Alice has implemented properly our security requirements on the large magnitudes of $Order_{\phi(n)}(2)$ and $Order_n(a)$. Then we observe that the mapping from a^e to $a^e(t)$ is random (which follows the Blum-Blum-Shub random sequence generator [2]) in a large subset of the quadratic residues modulo n. Thus, given the difficulty of extracting the e-th root of a random element in the RSA group, a successful extraction of $a(t)$ from $a^e(t)$ will constitute a grand breakthrough if it is done at a cost less than t squarings modulo n.

The above part of the argument (i.e., difficulty of finding $a(t)$ from $a^e(t)$) will also apply to the security analysis in §3.3.3.

Next, we observe that our scheme for encrypting $M \in Z_n^*$ inside $TE(M,t)$ is a trapdoor one-way permutation (from Z_n^* to a subset of it) since the transformation is to multiply, modulo n, the message M to the trapdoor secret $a(t)$. Thus, well-known plaintext ranomisation schemes which have been proposed for achieving the semantic security for trapdoor-one-way-permutation-based cryptosystems (e.g., OAEP for RSA [1]) can be applied to our plaintext message before the permutation and thereby achieve the message confidentiality properties that such a randomisation scheme offers (against various passive or active attacks).

3.3.2 Unforgeability of M^d in $TS(M,t)$

Recall that M here denotes an output from a secure one-way hash function before signing in the RSA way. The unforgeability of M^d in $TS(M,t)$ directly follows that of $M^d (\bmod\, n)$ given in clear.

Likewise, the randomness of $a^e(t)$ ensures that of $TS(M,t)^e$. Thus the availability of the pair $(TS(M,t), TS(M,t)^e)$ does not constitute a valid signature of Alice on anything since this availability is equivalent to that of (x, x^e) which can be constructed by anybody out of using a random x.

3.3.3 Indistinguishability of M^d in $TS(M,t)$

The indistinguishability is the following property: with the timed-release signature on M available at hand and with the proven membership $a^e(t) \in L(a^e, t, n)$, but without going through t squarings mod n, Bob must not be able to show to a third party that the data he possesses form a signature of Alice on M. The holding of this property is shown below.

Let $\hat{M} \in J_+(n)$ be any message of Bob's choice (e.g., $\hat{M}^d$ becomes available to him from a different context). We have

$$TS(M,t) \equiv a(t)M^d \equiv a(t)\left(\frac{M}{\hat{M}}\right)^d \hat{M}^d \equiv \hat{a}\hat{M}^d (\bmod n).$$

So the third party faces to decide which of M^d or $\hat{M}^d$ is sealed in $TS(M,t)$. This boils down to deciding if $a(t) \in L(a,t,n)$ or $\hat{a} \in L(a,t,n)$ (both are in $J_+(n)$). Even by making $a(t)$ and $\hat{a}$ available to the third party (and hence M^d and $\hat{M}^d$ become available too), without having viewed the membership proof protocol run between Alice and Bob, a correct decision will form a grand breakthrough if it is done at a cost less than t squarings mod n. We should emphasise the following point: even though the availability of M^d and $\hat{M}^d$ allows one to recognise that the both to be Alice's valid signatures, without verifying the membership status, one is unable to tell if any of the two has any connection with $TS(M,t)$ at all.

4 Membership Proof with Safe-Prime-Structured Modulus

Let Alice have constructed her RSA modulus n with a safe-prime structure. This requires $n = pq$, $p' = (p-1)/2$, $q' = (q-1)/2$ where p, q, p' and q' are all distinct primes of roughly equal size. We assume that Alice has proven to Bob in zero-knowledge such a structure of n. This can be achieved via using, e.g., the protocol of Camenisch and Michels [4].[1]

Let $a \in Z_n^*$ satisfy

$$gcd(a \pm 1, n) = 1, \tag{12}$$

$$\left(\frac{a}{n}\right) = -1. \tag{13}$$

It is elementary to show that a satisfying (12) and (13) has the full order $2p'q'$. The following lemma observes a property of a.

[1]Due to the current difficulty of zero-knowledge proof for a safe-prime-structured RSA modulus, we recommend to use the protocol in section 5 which works with any odd composite modulus provided it is difficult to factor. Section 4 merely serves a preparation purpose for Section 5.

$SQ(a, x, y, n)$

Input Common: n: an RSA modulus with a safe-prime structure;
$a \in Z_n^*$: an element of the full-order $2p'q' = \phi(n)/2$ (so $a \not\equiv \pm 1 \pmod n$);
$x, y \in J_+(n)$: $x \not\equiv \pm y \pmod n$;
Alice: z: $x \equiv \pm a^z \pmod n$, $y \equiv \pm a^{z^2} \pmod n$;

1. Bob chooses at random $r < n$, $s < n$ and sends to Alice: $C \stackrel{def}{=} a^r x^s \pmod n$;
2. Alice sends to Bob: $R \stackrel{def}{=} C^z \pmod n$;
3. Bob accepts if $R \equiv x^r y^s \pmod n$, or rejects otherwise.

Figure 1: Building Block Protocol

Lemma 1 *Let n be an RSA modulus of a safe-prime structure and $a \in Z_n^*$ of the full order. Then for any $x \in Z_n^*$, either $x \in \langle a \rangle$ or $-x \in \langle a \rangle$.*
Proof It's easy to check $-1 \notin \langle a \rangle$. So $\langle a \rangle$ and the coset $(-1)\langle a \rangle$ both have the half the size of Z_n^*, yielding $Z_n^* = \langle a \rangle \cup (-1)\langle a \rangle$. Any $x \in Z_n^*$ is either in $\langle a \rangle$ or in $(-1)\langle a \rangle$. The latter case means $-x \in \langle a \rangle$. $\square$

4.1 A Building Block Protocol

Let Alice and Bob have agreed on n (this is based on Bob's satisfaction on Alice's proof that n has a safe-prime structure).

Figure 1 specifies a perfect zero-knowledge protocol for Alice to prove that for $a, x, y \in Z_n^*$ with n of a safe-prime structure, a of the full order, and $x, y \in J_+(n)$, they satisfy (note, $\pm$ below means either $+$ or $-$, but not both)

$$\exists z : \ x \equiv \pm a^z \pmod n, \qquad y \equiv \pm a^{z^2} \pmod n. \tag{14}$$

Alice should of course have constructed a, x, y to satisfy (14). She sends a, x, y to Bob.

Bob (has checked n of a safe-prime structure) should first check (12) and (13) on a for its full-order property (the check guarantees $a \not\equiv \pm 1 \pmod n$); he should also check $x, y \in J_+(n)$.
Remark For ease of exposition this protocol appears in a non zero-knowledge format. However, the zero-knowledge property can be added to it using the notion of a commitment function: Instead of Alice sending R in Step 2, she sends a commitment $commit(R)$, after which Bob reveals r and s; this allows Alice to check the correct formation of C; the correct formation means that Bob has already known Alice's response.
Theorem 1 *Let a, x, y, n be as specified in the common input in Protocol SQ. The protocol has the following properties:*
Completeness *There exists $z \in Z_n$ and $x, y \in Z_n^*$ satisfying (14); for these values Bob will always accept Alice's proof;*

Soundness *If (14) does not hold for the common input, then Alice, even computationally unbounded, cannot convince Bob to accept her proof with probability greater than* $\frac{2p'+2q'-1}{2p'q'}$.[2]
Zero-knowledge *Bob gains no information about Alice's private input.*
Proof
Completeness For any $z \in Z_n$, let $x = a^z (\bmod n)$, $y = a^{z^2} (\bmod n)$ (both in the plus case). It is evident from inspection of the protocol that Bob will always accept Alice's proof.
Soundness Suppose that (14) does not hold whereas Bob has accepted Alice's proof.

The first congruence of (14) holds as a result of Lemma 1. So it is the second congruence of (14) that does not hold. Let $\xi \in Z_n^*$ satisfy

$$y \equiv \xi a^{z^2} (\bmod n) \quad \text{with} \quad Order_n(\xi) > 2. \tag{15}$$

By asserting $Order_n(\xi) > 2$ we exclude the cases for ξ being any square root of 1, which consists of either ± 1, or the other two roots which will render $y \notin J_+(n)$.

We only need to consider the case $x \equiv -a^z (\bmod n)$. The other case $x \equiv a^z (\bmod n)$ is completely analogous (and easier).

Since Bob accepts the proof, he sees the following two congruences

$$C \equiv a^r x^s (\bmod n), \tag{16}$$

$$R \equiv x^r y^s \ (\bmod n). \tag{17}$$

Examining (16), we see that $C \equiv a^r(-x)^s \in \langle a \rangle$ if s is even, or $-C \equiv a^r(-x)^s \in \langle a \rangle$ if s is odd. So for either cases of s, we are allowed to re-write (16) into the following linear congruence with r and s as unknowns

$$\log_a \pm C \equiv r + sz \, (\bmod 2p'q').$$

For every case of $s = 1, 2, \cdots, 2p'q'$, this linear congruence has a value for r. This means that for any fixed C, (16) has exactly $2p'q'$ pairs of solutions. Each of these pairs will yield an R from (17). Below we argue that for any two solution pairs from (16), which we denote by (r, s) and (r', s'), if $gcd(s - s', 2p'q') \le 2$ then they must yield $R \not\equiv R' (\bmod n)$. Suppose on the contrary

$$a^r x^s \equiv C \equiv a^{r'} x^{s'} (\bmod n), \quad \text{i.e.,} \quad a^{r-r'} \equiv x^{s'-s} (\bmod n), \tag{18}$$

it also holds

$$x^r y^s \equiv R \equiv R' \equiv x^{r'} y^{s'} (\bmod n), \quad \text{i.e.,} \quad x^{r-r'} \equiv y^{s'-s} (\bmod n). \tag{19}$$

Using (18) and (15) with noticing $x \equiv -a^z$, we can transform (19) into

$$(-1)^{[r-r'+z(s'-s)]} a^{[z^2(s'-s)]} \equiv x^{r-r'} \equiv y^{s'-s} \equiv \xi^{(s'-s)} a^{[z^2(s'-s)]} (\bmod n),$$

[2] The safe-prime structure of n implies $p' \approx q' \approx \sqrt{n}$ and hence this probability value is approximately $1/\sqrt{n}$.

which yields

$$\xi^{(s'-s)} \equiv (-1)^{[r-r'+z(s'-s)]} \equiv \pm 1 \,(\bmod\, n), \quad \text{i.e.,} \quad \xi^{2(s'-s)} \equiv 1 \,(\bmod\, n). \tag{20}$$

Recall that $Order_n(\xi) > 2$ which implies $Order_n(\xi)$ being a multiple of p' or q' or both. However, $gcd(s' - s,\, 2p'q') \leq 2$, i.e., $gcd(2(s' - s),\, 2p'q') = 2$, so $2(s' - s)$ cannot be such a multiple. Consequently (20) cannot hold and we reach a contradiction.

For any $s \leq 2p'q'$, it's routine to check that there are $2p' + 2q' - 2$ cases of s' satisfying $gcd(2(s' - s),\, 2p'q') > 2$. Thus, if (14) does not hold, amongst $2p'q'$ possible R's matching the challenge C, there are in total $2p' + 2q' - 1$ of them (matching s and the other $2p' + 2q' - 2$ s's) that may collide to Bob's fixing of R. Even computationally unbounded, Alice will have at best $\frac{2p'+2q'-1}{2p'q'}$ probability to have responded correctly.

Zero-Knowledge Immediate (see Remark after the description of the protocol). □

4.2 Proof of Membership in $L(a, t, n)$

For $t \geq 1$, we can express 2^t as

$$2^t = \begin{cases} 2^{[2\cdot(t/2)]} = [2^{(t/2)}]^2 & \text{if } t \text{ is even} \\ 2^{[2\cdot(t-1)/2+1]} = [2^{(t-1)/2}]^2 \cdot 2 & \text{if } t \text{ is odd} \end{cases}$$

Copying this expression to the exponent position of $a^{2^t}(\bmod\, n)$, we can express

$$a^{2^t}(\bmod\, n) \equiv \begin{cases} a^{[2^{(t/2)}]^2} & \text{if } t \text{ is even} \\ (a^{[2^{(t-1)/2}]})^2 & \text{if } t \text{ is odd} \end{cases} \tag{21}$$

In (21) we see that the exponent 2^t can be expressed as the square of another power of 2 with t being halved in the latter. This observation suggests that repeatedly using SQ, we can demonstrate, in $\lfloor \log_2 t \rfloor$ steps, that the discrete logarithm of an element is of the form 2^t. This observation translates precisely into the protocol specified in Figure 2 which will terminate within $\log_2 t$ steps and prove the correct structure of $a(t)$. The protocol is presented in three columns: the actions in the left column are performed by Alice, those in the right column, by Bob, and those in the middle, by the both parties.

A run of $Membership(a, t, a(t), n)$ will terminate within $\lfloor \log_2 t \rfloor$ loops, and this is the completeness property. The zero-knowledge property follows that of SQ. We only have to show the soundness property.

Theorem 2 *Let $n = (2p' + 1)(2q' + 1)$ be an RSA modulus of a safe-prime structure, $a \in Z_n^*$ be of the full order $2p'q'$, and $t > 1$. Upon acceptance termination of $Cert\_Est(a, t, a(t), n)$, relation $a(t) \equiv a^{2^t}(\bmod\, n)$ holds with probability greater than*

$$1 - \frac{\lfloor \log_2 t \rfloor (2p' + 2q' - 1)}{2p'q'}.$$

$Membership(a, t, a(t), n)$

Abort and reject if any checking by Bob fails, or accept upon termination.

Alice		Bob
	$u \stackrel{def}{=} a(t);$	$u \stackrel{?}{\in} J_+(n);\ a \stackrel{?}{\not\equiv} \pm u \pmod n$
While $t > 1$ do		
$y \stackrel{def}{=} u;$		
if t is odd: $y \stackrel{def}{=} a(t-1);$		
$x \stackrel{def}{=} a(\lfloor t/2 \rfloor);$		
Sends x, y to Bob;		
		Receives x, y from Alice;
		$x, y \stackrel{?}{\in} J_+(n);$
		if t is odd: $y^2 \stackrel{?}{\equiv} u \pmod n;$
	$SQ(a, x, y, n);$	
	$u \stackrel{def}{=} x;$	
	$t \stackrel{def}{=} \lfloor t/2 \rfloor;$	
		When $t = 1$:
		$u \stackrel{?}{\equiv} a^2 \pmod n;$

Figure 2: Membership Proof Protocol

Proof Denote by $SQ(a, x_1, y_1, n)$ and by $SQ(a, x_2, y_2, n)$ any two consecutive acceptance calls of SQ in $Membership$ (so $y_1 = a(t)$ in the first call, and $x_2 = a^2$ in the last call, of SQ in $Membership$, respectively). When $t > 1$, such two calls prove that there exists z:

$$x_2 \equiv \pm a^z \pmod n, \quad y_2 \equiv \pm a^{z^2} \pmod n, \tag{22}$$

and either

$$x_1 = y_2 \equiv \pm a^{z^2} \pmod n, \quad y_1 \equiv \pm a^{z^4} \pmod n, \tag{23}$$

or

$$x_1 = y_2^2 \equiv a^{2z^2} \pmod n, \quad y_1 \equiv \pm a^{4z^4} \pmod n. \tag{24}$$

Upon $t = 1$, Bob further sees that $x_2 = a^2$. By induction, the exponents z (resp. z^2, z^4, $2z^2$, $4z^4$) in all cases of $\pm a^z$ (resp. $\pm a^{z^2}$, $\cdots$) in (22), (23) or (24) contain a single factor: 2, and the minus symbol disappears from (22), (23) and (24) since the even exponents imply all cases of x and y to be quadratic residues. So we can write $a(t) = a^{2^u} \pmod n$ for some natural number u.

Further note that each all of SQ causes an effect of having 2^u square-rooted in the integers which is equivalent to having u halved in the integers. Thus, exactly $\lfloor \log_2 u \rfloor$ calls (and no more) of SQ can be made. But Bob has counted $\lfloor \log_2 t \rfloor$ calls of SQ, therefore $u = t$.

Each acceptance call of SQ has the correctness probability of $1 - \frac{2p'+2q'-1}{2p'q'}$. So after $\lfloor \log_2 t \rfloor$ acceptance calls of SQ, the probability for $Membership$ to be correct is

$$(1 - \frac{2p' + 2q' - 1}{2p'q'})^{\lfloor \log_2 t \rfloor} > 1 - \frac{\lfloor \log_2 t \rfloor (2p' + 2q' - 1)}{2p'q'}. \qquad \square$$

Discussions

i) It is obvious that by preparing all the intermediate values in advance, $Membership$ can be run in parallel to save the $\lfloor \log_2 t \rfloor$ rounds of interactions.

ii) In our applications described in §3, we will always prove $a^e(t) \in L(a^e, t, n)$ where e satisfies $gcd(e, \phi(n)) = 1$ (i.e., e is an RSA encryption exponent). Thus, a^e preserves the full order property to allow proper running of SQ and $Membership$.

iii) In case of proving the correctness of $a(t)$ with an intention for a reconstruction to be done in t squarings (e.g., reconstruction of $a(t-1)$ to be done in $t-1$ squarings), we should note that a run $Membership(a, t, a(t), n)$ has caused disclosure of $a(\lfloor t/2 \rfloor)$ for even t and $a(t-1)$ for odd t. This disclosure allows the reconstruction to be done in $t/2$ or 0 squarings, respectively. To compensate the loss of computation, proof of $a(2t)$ is necessary. Consequently, $Membership(a, 2t, a(2t), n)$ runs one loop more than $Membership(a, t, a(t), n)$ does. Note that this precaution is unnecessary for our applications in §3 because there it is the e-th root of the disclosed value that is needed but is not available still.

4.3 Performance

In each run of SQ, Alice (resp. Bob) performs one (resp. four) exponentiation(s) mod n. So in $Membership(a, t, a(t), n)$ Alice (resp. Bob) will perform $\lfloor \log_2 t \rfloor$ (resp. $4\lfloor \log_2 t \rfloor$) exponentiations mod n. These translate to $O(\lfloor \log_2 t \rfloor (\log_2 n)^3)$ bit operations.

In the LCS35 Time Capsule Crypto-Puzzle [10], $t = 79685186856218$ is a 47-bit binary number. Thus the verification for that puzzle can be completed within $4 \times 47 = 188$ exponentiations mod n.

The number of bits to be exchanged is measured by $O((\lfloor \log_2 t \rfloor)(\log_2 n))$.

5 Membership Proof with General Modulus

Now we show that our membership proof protocol can work with a modulus which is any odd composite integer provided it has two distinct prime factors (so factoring can be difficult). Our trick is to work with n^2 and prove

$$a(t) \in L(a, t, n^2)$$

where $a(t)$ is constructed modulo n^2 (to be specified in (25) and (26) below). Once the above is proven, $a(t) \pmod n \in L(a, t, n)$ results straightforwardly.

We begin by presenting a lemma which observes an interesting property of elements in $Z^*_{n^2}$ where n is any odd composite integer with at least two distinct prime factors. (Paillier

Protocol $SQ2(a, x, y, n)$

Input: Common: n: an odd composite integer with at least two distinct prime factors;
$a, x, y \in Z^*_{n^2}$: $x \not\equiv \pm a \pmod{n^2}$ and x is in the orbit of a;
Alice: z: $x \equiv a^z \pmod{n^2}$, $y \equiv a^{z^2} \pmod{n^2}$;

1. Bob chooses at random $r < n^2$, $s < n^2$, and sends to Alice: $C \stackrel{def}{=} a^r x^s \pmod{n^2}$;
2. Alice sends to Bob: $R \stackrel{def}{=} C^z \pmod{n^2}$ with a non-interactive proof $R \in \langle C \rangle$;
3. Bob accepts if $R \equiv x^r y^s \pmod{n^2}$, or rejects otherwise.

Figure 3: Modified Building-Block Protocol

used the same group to have constructed new public-key cryptosystems [9], which does not use our observation.)

Lemma 2 *Let n be any odd composite integer. For a randomly chosen integer $u \in Z^*_{n^2}$,*

$$Pr[\ n \ divides\ Order_{n^2}(u)\ ] \geq \frac{\phi(n)}{n}.$$

Proof See Appendix A.

5.1 Modified Membership Proof Protocol

Let Alice have constructed $a(t) \pmod{n^2}$. She can do so efficiently by the following two steps

$$u \stackrel{def}{=} 2^t \pmod{\phi(n)n}, \tag{25}$$

$$a(t) \stackrel{def}{=} a^u \pmod{n^2}. \tag{26}$$

The building-block protocol SQ will be modified into $SQ2$ in Figure 3 which allows Alice to prove that a common input tuple (a, x, y, n) satisfies

$$\exists z: \ x \equiv a^z \pmod{n^2} \quad \text{and} \quad y \equiv a^{z^2} \pmod{n^2} \tag{27}$$

The modified protocol will require $a \in Z^*_{n^2}$ to have an order divisible by n. By Lemma 2, if a is output from a pseudo random generator which is seeded with n and a publicly verifiable seed, then this will almost certainly be the case. This way of fixing a can be verified by Bob. Also, we assume that x is in the orbit of a (as will be clear in a moment, this will always be seen by Bob in his verification which applies $SQ2$). Of course, Bob should check $x \not\equiv \pm a \pmod{n^2}$ before engaging a verification run with Alice.

Remark Besides the use of n^2, $SQ2$ differs from SQ in Step 2 where Alice adds a proof of subgroup membership, which is very simple (see e.g., Stinson [12], pages 399-400) and can be made non-interactive.

We only have to prove the soundness property for $SQ2$.

Theorem 3 *Let a, x, y, n be as specified in the common input of Protocol $SQ2$. The protocol has the following properties soundness property:*

Soundness *If (27) does not hold for the common input values, then Alice cannot convince Bob to accept her proof with probability greater than $\frac{n-\phi(n)+1}{n}$.*[3]

Proof See Appendix A.

Replacing SQ with $SQ2$ and n with n^2, $Membership$ is modified straightforwardly to working with n^2. Upon acceptance, Bob sees that when $t = 1$, x has an initial value generated by a. By the soundness property of $SQ2$, y will have an initial value generated by a using a power of 2, which has been used as the value of x in a previous loop. By induction, this status ($x \in \langle a \rangle$) will be maintained as long as Bob has accepted each run of $SQ2$. Thus after $\lfloor \log_2 t \rfloor$ instances of acceptance of $SQ2$, the modified $Membership$ has a correctness probability greater than

$$1 - \frac{\lfloor \log_2 t \rfloor (n - \phi(n) + 1)}{n}.$$

Finally we should recap that Bob's acceptance of $a(t) \in L(a, t, n^2)$ implies his acceptance of $a(t) \pmod n \in L(a, t, n)$. The timed-release encryption and signature schemes in §3 should remain working with modulo n, rather than n^2.

5.2 Performance

In $SQ2$, the additional step for verifying the subgroup membership condition will require Bob to compute an additional modulo exponentiation, while Alice's load remains the same. So Bob will compute 5 modulo exponentiations mod n^2.

The use of a modulus of double size will result in a 8-fold increase in local computations. Thus, to prove (resp. verify) $a(t) \in L(a, t, n^2)$ using the modified membership proof protocol, Alice (resp. Bob) will perform $8(\lfloor \log_2 t \rfloor)$ (resp. $(5 \times 8)(\lfloor \log_2 t \rfloor)$) exponentiations mod n. (These measurements have been converted to the modulo n operation.)

6 Conclusion

We have constructed general and efficient cryptographic protocol schemes for achieving timed-release cryptography which include timed-release encryption and timed-release signatures. These schemes have proven correctness on time control which can be fine tuned to the granularity in the number of multiplications.

We have also shown that the use of n^2 can relax the structural requirement on n. This is an important observation which indicates that many RSA-based protocols which require the use of safe-prime structured moduli can be modified this way to working with standard moduli. Therefore this observation forms an independent contribution to the area of study.

[3]For n being a standard RSA modulus, i.e., product of two primes of roughly equal size, this probability value is $\approx 1/\sqrt{n}$.

Acknowledgments

I would like to thank Kenny Paterson and Steven Galbraith for their helpful comments on a draft of this paper.

References

[1] Bellare, M., Desai, A., Pointcheval, D. and Rogaway, P. Relations among notions of security for public-key encryption schemes, Advances in Cryptology: Proceedings of CRYPTO 98 (H. Krawczyk ed.), Lecture Notes in Computer Science 1462, Springer-Verlag 1998, pages 26-45.

[2] Blum, L., Blum, M. and Shub, M. A simple unpredictable pseudo-random number generator, SIAM J. Comput. 15(2): 364-383 (1986).

[3] Boneh, D. and Naor, M. Timed commitments (extended abstract), Advances in Cryptology: Proceedings of CRYPTO'00, Lecture Notes in Computer Science 1880, Springer-Verlag 2000, pages 236–254.

[4] Camenisch J. and Michels, M. Proving in zero-knowledge that a number is the product of two safe primes, In Advances in Cryptology — EUROCRYPT 99 (J. Stern ed.), Lecture Notes in Computer Science 1592, Springer-Verlag 1999, pages 106–121.

[5] Chaum, D. Zero-knowledge undeniable signatures, Advances in Cryptology: Proceedings of CRYPTO 90 (I.B. Damgaard, ed.) Lecture Notes in Computer Science 473, Springer-Verlag 1991, pages 458-464.

[6] Damgård, I. Practical and probably secure release of a secret and exchange of signatures, Advances in Cryptology — Proceedings of EUROCRYPT 93 (T. Helleseth ed.), Lecture Notes in Computer Science 765, Springer-Verlag 1994, pages 200–217.

[7] Gennaro, R., Krawczyk, H. and Rabin, T. RSA-based undeniable signatures, Advances in Cryptology: Proceedings of CRYPTO 97 (W. Fumy ed.), Lecture Notes in Computer Science 1294, Springer-Verlag 1997, pages 132-149. Also in *Journal of Cryptology* (2000)13:397–416.

[8] Goldreich, O., Micali, S. and Wigderson, A. How to prove all NP statements in zero-knowledge and a methodology of cryptographic protocol design, Advances in Cryptology — Proceedings of CRYPTO 86 (A.M. Odlyzko ed.), Lecture Notes in Computer Science, Springer-Verlag 263 (1987), pages 171–185.

[9] Paillier, P. Public-key cryptosystems based on composite degree residuosity classes, Advances in Cryptology — Proceedings of EUROCRYPT 99 (J. Stern ed.), Lecture Notes in Computer Science, Springer-Verlag 1592 (1999), pages 223–238.

[10] Rivest, R.L. Description of the LCS35 Time Capsule Crypto-Puzzle, `http://www.lcs.mit.edu/about/tcapintro041299`, April 4th, 1999.

[11] Rivest, R.L., Shamir, A. and Wagner, D.A. Time-lock puzzles and timed-release crypto, Manuscript. Available at (`http://theory.lcs.mit.edu/~rivest/RivestShamirWagner-timelock.ps`).

[12] Stinson, D.R. Cryptography: Theory and Practice, CRC Press, 1995.

[13] van Oorschot, P.C. and Wiener, M.J. Parallel collision search with cryptanalytic applications, *J. of Cryptology*, Vol.12, No.1 (1999), pages 1–28.

A Proofs

Lemma 2 *Let n be any odd composite integer. For a randomly chosen integer $u \in Z^*_{n^2}$,*

$$Pr[\ n\ divides\ Order_{n^2}(u)\ ] \geq \frac{\phi(n)}{n}.$$

Proof Write $n = \prod_{i=1}^r p_i^{e_i}$ with p_i (for $i = 1, 2, \cdots, r$) being distinct odd primes. Let $i = 1, 2, \cdots, r$.

For any $x \in Z^*_{n^2}$ denote by $x_i \in Z^*_{p_i^{2e_i}}$ the result of $x \bmod p_i^{2e_i}$. Then $x \in Z^*_{n^2}$ has an order divisible by n if and only if $x_i \in Z^*_{p_i^{2e_i}}$ has an order divisible by $p_i^{e_i}$, i.e., the order is $p_i^{e_i}k$ for $k|\phi(p_i^{e_i})$. In the cyclic group $Z^*_{\left(p_i^{2e_i}\right)}$, the number of elements of order $p_i^{e_i}k$ for $k|\phi(p_i^{e_i})$ is $\phi(p_i^{e_i}k)$. Summing them up for all the cases of k, the number of such elements in the $Z^*_{\left(p_i^{2e_i}\right)}$ is

$$\sum_{p_i^{e_i}k|\phi\left(p_i^{2e_i}\right)} \phi(p_i^{e_i}k) \ \geq\ \phi(p_i^{e_i}) \sum_{k|\phi(p_i^{e_i})} \phi(k) = \phi(p_i^{e_i})^2.$$

The inequality meets the equation case only when $gcd(\phi(n), n) = 1$ and thereby $\phi(p_i k) = \phi(p_i)\phi(k)$. Thus, in $Z^*_{n^2}$, the number of elements of orders divisible by n is at least

$$\prod_{i=1}^r \phi(p_i^{e_i})^2 = \phi(\prod_{i=1}^r p_i^{e_i})^2 = \phi(n)^2.$$

The claimed probability bound follows from the fact that $Z^*_{n^2}$ has $\phi(n)n$ elements. □

Theorem 3 *Let a, x, y, n be as specified in the common input of Protocol SQ2. The protocol has the following properties soundness property:*
Soundness *If (27) does not hold for the common input values, then Alice cannot convince Bob to accept her proof with probability greater than $\frac{n-\phi(n)+1}{n}$.*[4]

[4] For n being a standard RSA modulus, i.e., product of two primes of roughly equal size, this probability value is $\approx 1/\sqrt{n}$.

Proof Suppose that (27) does not hold whereas Bob has accepted Alice's proof. Since x is in the orbit of a, so it is the second congruence of (27) that does not hold. We can denote $z = \log_a x$ and

$$\exists \xi \neq 1 : y \equiv \xi a^{z^2} (\bmod\, n^2). \tag{28}$$

Since Bob accepts the proof, he sees the following two congruences (noticing (28) with $x \equiv a^z$):

$$C \equiv a^r x^s \equiv a^{r+sz} (\bmod\, n^2), \tag{29}$$

$$R \equiv x^r y^s \equiv a^{(r+sz)z} \xi^s \equiv C^z \xi^s (\bmod\, n^2).$$

Since Alice has also proven $R \equiv C^k (\bmod\, n^2)$ for some k, we derive

$$C^{k-z} \equiv \xi^s (\bmod\, n^2). \tag{30}$$

On the other hand, in (29), $\log_a C \in \langle a \rangle$ since $x \in \langle a \rangle$, so writing $Order_{n^2}(a) = \ell n$ for some integer $\ell | \phi(n)$, we are allowed to rewrite (29) into the following linear congruence

$$\log_a C \equiv r + sz\, (\bmod\, \ell n).$$

For each case of $s = 1, 2, \cdots, \ell n$, this linear congruence has a value for r, and so it has exactly ℓn distinct solution pairs. Note that these pairs are solved from the fixed C, a, x, and so they are independent from k and the fixed z. So the right hand side of (30) is a constant for all cases of $s = 1, 2, \cdots, \ell n$; in particular, for the cases of $s = 1, 2$, we have:

$$1 \equiv \xi^{2-1} \equiv \xi\, (\bmod\, n^2).$$

This contradicts (28).

Since we derive the contradiction on the condition that $R \in \langle C \rangle$, the probability for Alice's successful cheating is therefore the same as that for $R \notin \langle C \rangle$, i.e., the error probability of the subgroup membership proof (in Step 2). If $Order_{n^2}(C)$ is a multiple of n, then the latter probability is bounded by $1/n$. Thus, using the result of Lemma 2, we have (note that $Pr[E|F]$ denotes the conditional probability)

$$\begin{aligned} Pr[\text{Alice Cheats}] &= Pr[R \notin \langle C \rangle | Order_{n^2}(C) \geq n] Pr[Order_{n^2}(C) \geq n] + \\ &\quad\; Pr[R \notin \langle C \rangle | Order_{n^2}(C) < n] Pr[Order_{n^2}(C) < n] \\ &< 1/n + 1 - \phi(n)/n = \frac{n - \phi(n) + 1}{n}. \quad \square \end{aligned}$$